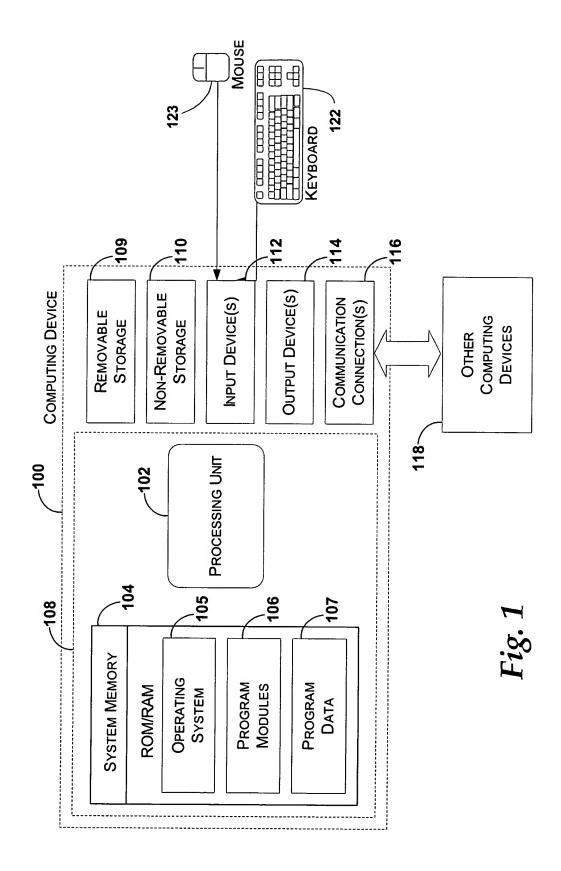
Inventor: Voellm et al.

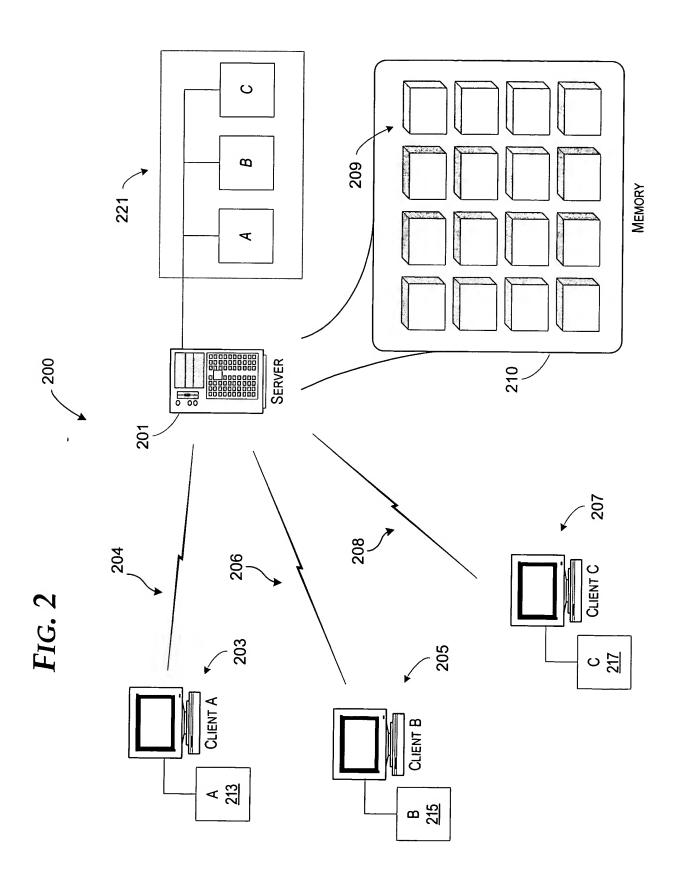
į.

Docket No.: 50037.188US01
Title: SYSTEM AND METHOD FOR DYNAMICALLY ALLOCATING RESOURCES IN A CLIENT/SERVER ENVIRONMENT



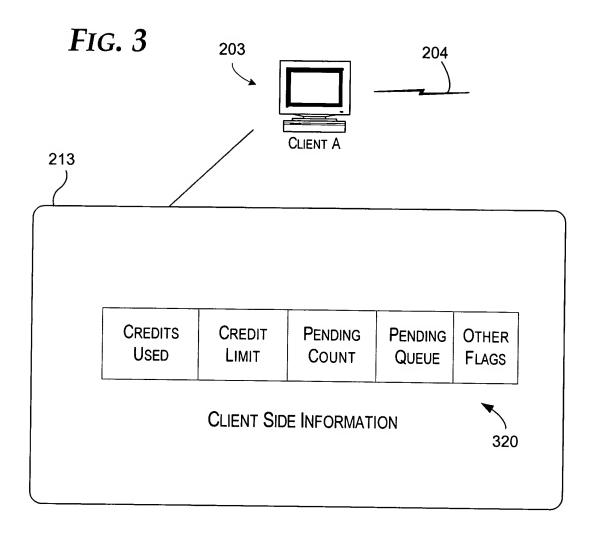
Title: SYSTEM AND METHOD FOR DYNAMICALLY ALLOCATING

RESOURCES IN A CLIENT/SERVER ENVIRONMENT



fitle: SYSTEM AND METHOD FOR DYNAMICALLY ALLOCATING

RESOURCES IN A CLIENT/SERVER ENVIRONMENT



Title: SYSTEM AND METHOD FOR DYNAMICALLY ALLOCATING

RESOURCES IN A CLIENT/SERVER ENVIRONMENT

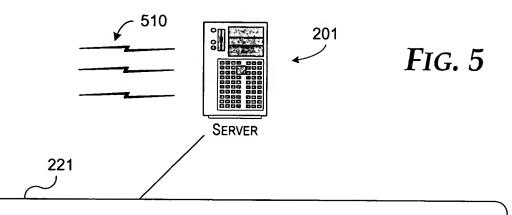
FIG. 4

Negotiate initial minimum credit allocation called the Credit Limit

```
While (forever) {
                    Wait for a new request, completion or credit message
                    If (new request) {
                            If (Current Used Credits < Credit Limit) {
403
                                   Update Current Used Credits by +1
                                   Insert number of pending I/O's into the current request
                                   Send new request to the server
                           } else {
                                   Queue new request to the pending queue
                    } else if (completion) {
                           Complete I/O
                           If (uses posted receives) {
                                   Repost receive
                           If (request pending) {
                                   Remove request from pending queue and send to the sever.
                           } else {
                                   Update Current Used Credits by -1
                   } else /* credit message */ {
                           if (plus N credit message) {
                                   if (uses posted receives) {
407
                                          Repost receive
                                   Dequeue and send N pending messages with
                                          one marked as a credit ack
                                  Update Current Used Credits by the actual number sent
                                  Update Credit Limit by N
                           } else if (minus N credit message) {
                                  if (uses posted receives) {
                                          Repost Receive
409
                                  Update Credit Limit by -N
                                  If (flushing posted receives) {
                                          Send N credit messages to the sever up to Credit Limit
                                          Queue unsent credits messages to the pending queue
                          } if (uses posted receives) {
                            Fr e p sted receive
                   }
```

Title: SYSTEM AND METHOD FOR DYNAMICALLY ALLOCATING

RESOURCES IN A CLIENT/SERVER ENVIRONMENT



SERVER SIDE INFORMATION

CLIENT A

CREDITS USED CREDIT LIMIT	PENDING COUNT		OTHER FLAGS	
-----------------------------	------------------	--	----------------	--

CLIENT B

CREDITS USED	CREDIT LIMIT	PENDING COUNT		OTHER FLAGS
--------------	--------------	------------------	--	----------------

CLIENT C

CREDITS USED	CREDIT LIMIT	PENDING COUNT		OTHER FLAGS
--------------	--------------	------------------	--	----------------

Title: SYSTEM AND METHOD FOR DYNAMICALLY ALLOCATING

RESOURCES IN A CLIENT/SERVER ENVIRONMENT

```
I* i is the index for a particular client instance. */
/* i,j is the index for file j on client instance i */
                                                                                           FIG. 6
          While (forever) {
                     Wait for new client connection or
                     new client I/O request or
                     credit timer expires or
                     credit message
                     if (new connection) {
                               Process Connection Request
                               Negotiate minimum credits for the connection
  603
                               Accept connection if credits available otherwise reject
                               Set Credit Timer if connection accepted.
                    } else If (new client I/O request) {
                               /* CP = Client hint on number Pending requests to be sent */
                               If (CP of new request > CP<sub>i</sub>) then {
                                          Store CP in CP
                                          Set Credit Timer if not already set
  605
                               Dispatch client I/O request
                    } else if (credit message) {
                               Acknowledge credit message
 607
                               If all credits acknowledge then mark connection as acknowledged
                    } else if (credit timer expired) {
                               Compute; {
                                         Total Client Connections (TC) = \sum_{i=1}^{n} 1
                                         Total Files open (TF) = \sum_{i,j} 1
                                         Total pending requests (TR) = \sum_{i} CP_{i}
                                         /* CU<sub>i</sub> is the number of credits in use on connection i */
                                         Total credits used (TU) = \sum_{i} CU_{i}
                               for each client connection i {
                                         if (connection i waiting for ack) {
                                                    do nothing or disconnect if wait time expired
                                                    if ((TU + TR) > MC) {
                                                               if (be fair to files) {
                                                                         NCL; = MC * CF/TF
  609
                                                              } else /* be fair to connections */ {
                                                                         NCL = MC/TC
                                                    } else {
                                                               NCL<sub>i</sub> = infinite;
                                                    }
                                                    Adjust NCL<sub>i</sub> to meet minimum negotiated
                                                    if (CL<sub>i</sub> > NCL<sub>i</sub>) then {
                                                               Send minus delta credit message of NC<sub>i</sub> - NCL<sub>i</sub>
                                                    } else if (((CL_i * Completion Factor) – CU_i) > 0) then {
                                                               Send minus delta credit message of CL, - CU,
                                                    } else if (CP_1 > 0) {
                                                              Send positive delta credit message of
                                                                        min(CP<sub>i</sub>, NCL<sub>i</sub> - NC<sub>i</sub>, MC - TU)
                                                                        increase TU by N and reduce TR by N
                                                    }
                                        }
                    }
         }
```